TRIGGER-HAPPY AUDITING

THIS MONTH OUR INDEPENDENT SOFTWARE TESTER **DAVE GEORGE** INVESTIGATES COSYN SOFTWARE'S AUDIT TRAIL/400, A TOOL HE SAYS WAS JUST WAITING TO BE INVENTED

Auditing is a strange area of iSeries application design. Strange in that it always seems to be an afterthought, and even stranger in the methods used to deploy it. In accounting, an audit trail is the sequence of paperwork that validates or invalidates accounting entries. In computing, the term is also used for an electronic or paper log used to track computer activity.

For example, a corporate employee might have access to a section of a network in a corporation such as billing, but be unauthorised to access all other sections. If that employee attempts to access an unauthorised section by typing in passwords, this improper activity is recorded in the audit trail. And this type of activity is commonly attributed to systems security auditing, to ensure that access to areas of the system is not mistreated.

There is often a requirement for an even lower level of auditing, that of recording events pertaining to the activity applied to the application database. The data audit requirement would normally be to record transactions made on certain key files that will be able to trace changes from the origin of the data to its current position. Traditionally, auditing capabilities on iSeries applications have been added to the application by either building audit files into the application, or by using journals to record each changed record. Each of these methods had their own problems.

Using the application to record data changes internal to the application meant that changes made outside of the application just didn't get recorded. Using journals to record changed data offers more than just auditing capabilities, but requires that the whole record be saved and does not offer field level auditing. Journaling also manages to bring on a cold sweat at the mere mention of its name.

With the arrival of DB2 triggers, a method of file auditing was introduced that provided just what the audit doctor ordered – an auditing method bound to the file that is invoked whenever the trigger criteria is met independent of the application, and configurable using a trigger program that can record specific fields and not whole records. Of course, triggers were never embraced in the way that IBM intended, and may have had more of a frightening effect than journals!

BRIGHT SPARKS

The auditing method was there, the auditing requirement was there, all that was needed was for some bright sparks to tie it all together and produce a simple, structured, comprehensive management application to plug the gap. Enter Audit Trail/400.

The developers at New Zealand-based Cosyn Software Limited are the authors of Audit Trail/400, which strives to provide a comprehensive audit trail and reporting mechanism using only a handful of set-up requirements.

Before jumping into the product, it is worth appreciating what requirements the product intends to fulfil. Audit Trail/400 is a specific optimised suite of programs to provide a method of recording changes to critical master files whose data content requires control. Ideally, this is best suited to record changes made to critical master files. It also gives a certain edge over recording changes made to transactional data via journals, in that it allows you to record just the data fields you

wish to capture.

As with all good utilities, the product strives to remove the complexity and facilitate easier and standard access via tools for managing the auditing of data; careful consideration is required in building the plan for auditing your shop's data. Therefore, an understanding of the framework of the software applications involved, their interaction with the database, as well as clear, firm requirements for auditing, are mandatory for achieving success.

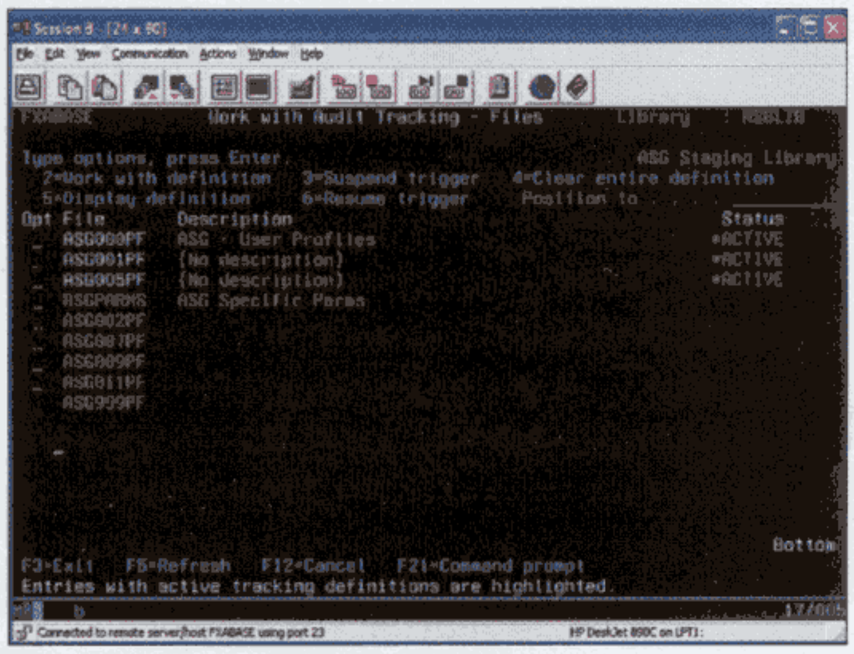

FIGURE 1

Operationally, the system is extremely simple to use and requires only a minimal number of choices to establish your trail (see Figure 1). For example, Select Library, Select File to use, Select key fields from file, Select fields to Audit and so on. The only other addition to the set-up is to choose which audit fields that you require Audit Trail/400 to add to the audit records, such as date/time, user, job, program, member etc, to give body to the audit trail.

Interestingly, only libraries that are in the user portion of the library list and that are not system libraries (ie not prefixed with 'Q') are available for selection, which requires a user library to be manually added to the job's library list if it does not currently exist. While running Audit Trail/400, the COSYN product library is not available for audit (obviously). Any user libraries that may be in the system portion of the library list are not available either, but this will be added to a future release of the product.

The main requirement for creating an audit trail is that the physical file must be externally described and contain at least two fields, one of which will be defined as a key field and the other as a field to be audited. Once the audit information has been specified, a trigger is built from the specified definition and applied to the selected file by an Audit Trail/400 batch function. From now on, any updates to the selected file affecting a traced field will result in the change being added into the audit trail file created by Audit Trail/400. The audit definition can be managed accordingly; either suspended, resumed, cleared, deleted or changed. Each change invokes a new version of the audit files, so that existing data may be retained.

While this seems a flawless operation so far, there are a few additional caveats. Multi member files are supported within Audit Trail/400 but, by the current definition of a trigger, it is applied to the whole file and not just a particular member. Audit Trail/400 handles this by allowing the definition to record all members, a single member or selected members. This is defined in the trail definition.

Currently (with V5R1), only 300 triggers can be added to a single physical file. This means that Audit Trail/400 will happily coexist with a file that has triggers for other applications, up to the allowed maximum. While it is unlikely that a master file will already have so many triggers, Audit Trail/400 will issue a warning, indicating the implications of adding a trigger to an already triggered physical file. Prior to V5R1, a physical file was only allowed one 'before' and one 'after' the trigger program. To get round this, an API call is provided for hooking into the Audit Trail/400 trigger program from your own trigger wrappers.

Triggers cannot be added to a file that is in use. If you attempt to add a trail to an in-use file you are presented with a list of object locks, and an option to have the trigger submitted later when the file is unlocked.

At this point we begin to look at what we are generating within the audit trail. The initial

'AS WITH ALL GOOD UTILITIES, THE PRODUCT STRIVES TO REMOVE THE COMPLEXITY AND FACILITATE EASIER AND STANDARD ACCESS VIA TOOLS FOR MANAGING THE AUDITING OF DATA'

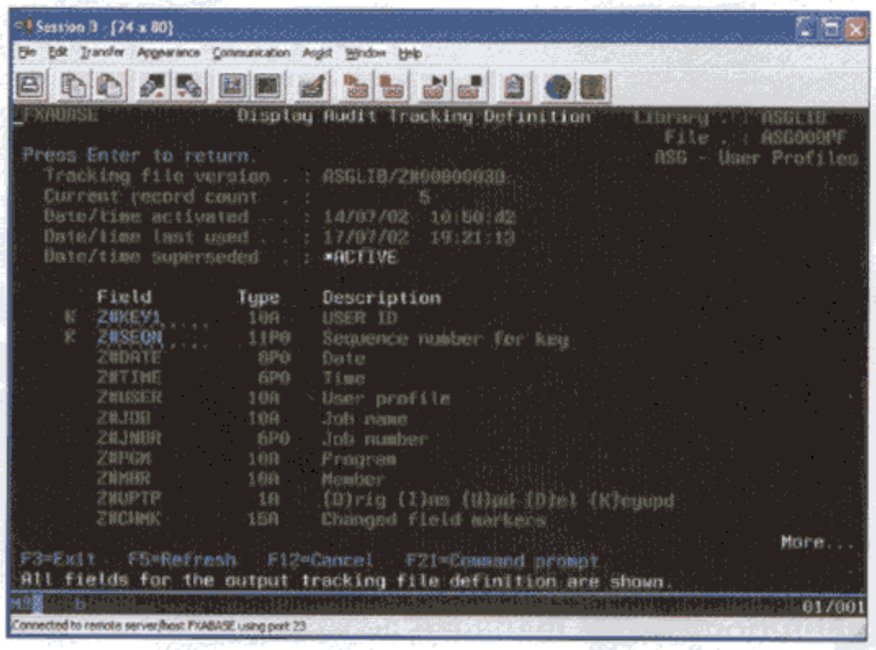

FIGURE 2

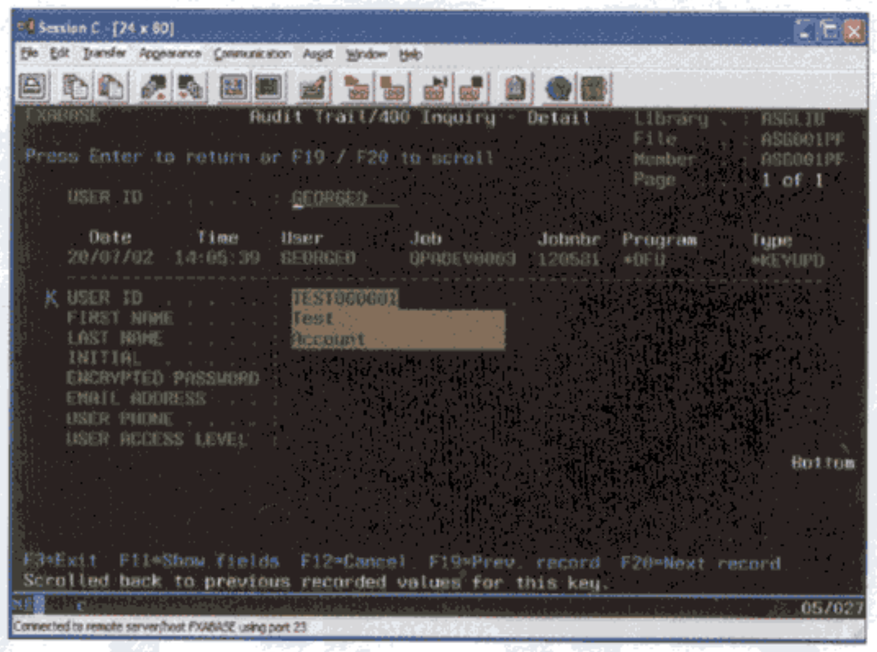

FIGURE 4

build of the audit trail file, created by adding a trigger to the file, builds a database file with a file definition that is equal to the required audit fields (user, time, job etc.) and the tracked fields from the original physical files. These field attributes are derived from the original files, and therefore are a true representation of the data, and not data strings that require analysis using additional programs or data structures (see Figure 2).

The file data is in an unencrypted data format; it is open to modification via the same methods that may have updated the original data – for example, a program, DFU, SQL etc. I would have preferred to see the data stored in encrypted form and only visible via Audit Trail/400 functions. However, by having the data available to HLL programs, SQL and Query/400, it means that the reproduction of audit data can be presented in any number of ways to suit any corporation's requirements.

Once a physical file is being tracked, the trail data can be reproduced by either online displays or by standard reports. Both reports and displays have a highly configurable feel to them, allowing you to select various filters and sorts, as well as which fields to display, providing a clean, consistent view into the change activity within a data set (see Figures 3 and 4).

The audit track files created by Audit Trail/400 reside in the same library as the original data file, so that in the case of a restore being required, the audit file and tracked version can be restored simultaneously. This illustrates just one of many of the steps taken within Audit Trail/400 to ensure the integrity of the data capture. However, increased awareness of the existence of the audit files in production libraries is necessary to ensure that they are not deleted.

DEEPLY SATISFYING

Minor criticisms aside, Audit Trail/400 does provide seamless (V5R1 and higher) and near seamless data auditing (pre-V5R1) in a consistent, simple-to-manage fashion, with easy-to-read, configurable tracing tools which would satisfy many auditing requirements. It provides an optimised data capture solution, while shielding administrators from a lot of the complexity of database triggers. With just an understanding of the way triggers impact the software, and vice versa, an administrator could effortlessly track data, without the overhead of adding many custom auditing routines, and without studying the ins and outs of triggers separately.

While testing was carried out on simple files with a small data set using SQL and DFU to issue updates, it was clear to see how a few simple actions could be easily deployed across a whole mountain of enterprise data. Although no immediate performance degradation was detected by the introduction of the DB trigger, larger-scale testing would be necessary to assess its impact.

The most prominent quality of an auditing tool of this stature is that it gives you a basis of consistent auditing of all applications deployed on your system, regardless of whether they are in-house or third party applications. Ever seen an auditor jump with joy? Who knows, you just might!

More details and a time-limited demo can be obtained by visiting www.cosynsoftware.com.

DAVE GEORGE IS THE EDITOR OF *400 TIMES*
WWW.400TIMES.CO.UK

FIGURE 3